UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/069,229	12/11/2002	Gilbert Wolrich	10559-305US1	1429

20985 7590 12/13/2006

FISH & RICHARDSON, PC
P.O. BOX 1022
MINNEAPOLIS, MN 55440-1022

EXAMINER	
JOO, JOSHUA	

ART UNIT	PAPER NUMBER
2154	

DATE MAILED: 12/13/2006

Please find below and/or attached an Office communication concerning this application or proceeding.

UNITED STATES PATENT AND TRADEMARK OFFICE

Commissioner for Patents
United States Patent and Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450
www.uspto.gov

BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

Application Number: 10/069,229
Filing Date: December 11, 2002
Appellant(s): WOLRICH ET AL.

Ido Rabinovitch
For Appellant

EXAMINER'S ANSWER

This is in response to the appeal brief filed 9/14/2006 appealing from the Office action mailed 5/24/2006.

(1) Real Party in Interest

A statement identifying by name the real party in interest is contained in the brief.

(2) Related Appeals and Interferences

The examiner is not aware of any related appeals, interferences, or judicial proceedings which will directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

(3) Status of Claims

The statement of the status of claims contained in the brief is correct.

(4) Status of Amendments After Final

The appellant's statement of the status of amendments after final rejection contained in the brief is correct.

(5) Summary of Claimed Subject Matter

The summary of claimed subject matter contained in the brief is correct.

(6) Grounds of Rejection to be Reviewed on Appeal

The appellant's statement of the grounds of rejection to be reviewed on appeal is correct.

(7) Claims Appendix

The copy of the appealed claims contained in the Appendix to the brief is correct.

(8) Evidence Relied Upon

4,724,521	Carron et al.	2-1988
4,742,451	Bruckert et al.	5-1988
5,202,972	Gusefski et al.	4-1993
5,748,950	White et al.	5-1998
5,802,373	Yates et al.	9-1998
5,898,866	Atkins et al.	4-1999
6,139,199	Rodriguez	10-2000

(9) Grounds of Rejection

The following ground(s) of rejection are applicable to the appealed claims:

Claim Rejections - 35 USC § 103

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

Claims 1-3, 6, 10, 13-21 are rejected under 35 U.S.C. 103(a) as being unpatentable over Carron et al, US Patent #4,724,521 (Carron hereinafter), in view of Yates et al, US Patent #5,802,373 (Yates hereinafter) and White et al, US Patent #5,748,950 (White hereinafter).

As per claim 1, Carron teaches substantially the invention as claimed including the method of operating a processor, Carron's teachings comprise of:

executing a branch instruction that causes a processor to branch from executing a first sequential series of instructions to a different sequential series of instructions (Col 134, lines 31-32. Process is branched if the test passes or continues with operation if the test fails.) based on a byte specified by an instruction in a buffer, being equal or not equal to a specified byte value (Col 134, lines 30-31. Compares byte values to a byte value.), if the specified byte matches or mismatches the byte value (Col 134, lines 31-33. Test passes or fails.).

Carron teaches substantial features of the claimed invention including instructions for comparing a byte in a register with a specified value (Col 39, lines 25-28), comparing a byte in a buffer with a specified byte value and performing branch instruction based on the comparison. However, Carron does not specifically teach a branch instruction that causes a processor to compare and branch based on a byte specified by the branch instruction; and of comparing a byte specified by the branch in instruction in a register and performing branching instructions based on the comparison.

White teaches of a processor executing a single instruction for performing compare and branch operations, wherein the branch/compare instruction specifies the sources for comparison. (Col 5, lines 25-43).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify the teachings of Carron with the teachings of White with the motivation that the teachings of White for a processor instruction to perform both compare and branch operations, wherein the instruction specifies the sources for comparison would improve the efficiency of Carron's system by reducing the number of the instructions executed by the processor.

Yates teaches of executing branch instructions based on comparing a byte in a register with a specified value (Col 77, lines 29-33).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teachings of Carron, White, and Yates with the motivation that the teachings of Yates to compare a byte value in a register and branch based on the comparison would improve the teachings of Carron by allowing the execution of instructions based on comparison of bytes values located not only in the buffer, but in other locations of the terminal.

As per claims 14, 17, and 20, Carron teaches substantially the invention as claimed including the method, processor, and program product for operating a processor, Carron's teachings comprise of:

executing a branch instruction by (Col 134, lines 8-12. Branch.):

a register stack (Col 39, lines 25-28; Col 40, lines 43-45. Register.).

an arithmetic logic unit coupled to the register stack and a program control store that stores a branch instruction that cues the processor to (Fig 3; Col 18, lines 10-15. Processor executes instructions. ALU is inherent. Col 7, lines 59-64. Instructions are stored in memory.):

fetch a byte stored in a buffer (Col 134, line 8. Byte from buffer.);

determine whether the byte in the buffer is equal or not equal to a specified byte value contained in the instruction (Col 134, lines 8-12. Compare byte with the value stored in the specified variable.); and

perform a branch operation specified by the branch instruction based on the specified byte being equal or not equal to the byte in the buffer (Col 134, lines 8-12. Branch if test passes.).

Carron teaches substantial features of the claimed invention including comparing a byte in a register with a specified value (Col 39, lines 25-28); and comparing a byte in a buffer with a specified byte value and performing a branch instruction based on the comparison (Col 134, lines 30-31). However, Carron does not specifically teach of fetching a byte, specified by the branch instruction, stored

in a register, specified by the branch instruction, and comparing the byte located in the register with a specified value to perform branching instructions.

White teaches of a processor executing a single instruction for performing compare and branch operations, wherein the branch/compare instruction specifies the sources for comparison. (Col 5, lines 25-43).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify the teachings of Carron with the teachings of White with the motivation that the teachings of White for a processor instruction to perform compare and branch operations, wherein the instruction specifies the sources for comparison would improve the efficiency of Carron's system by reducing the number of the instructions executed by the processor.

Yates teaches of executing branch instructions based on comparing a byte in a register with a specified value (Col 77, lines 29-33).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teachings of Carron and Yates because both teachings are similar in that they teach of executing branch instructions based on comparison of byte values. The teachings of Yates to compare a byte value in a register and branch based on the comparison would improve the teachings of Carron by allowing the terminal of Carron to execute instructions based on comparison of bytes values located not only in the buffer, but in other regions of the terminal.

As per claims 2, 15, 18, and 21, Carron teaches the invention wherein performing the branch instruction that causes the processor to perform the branching operation causes the process to branch to an instruction at a specified label (Col 118, lines 11-12. Branch to address label.).

As per claims 3, 16, and 19, Carron teaches the invention wherein branch instruction comprises: a bit_position field that specifies the byte in a longword contained in the register (Col 22, lines 39-52, Col 134, line 14-26. Indicates byte position, indicated by opcode commands.).

As per claim 6, Carron teaches the method of claim 1 wherein the register is a context-relative transfer register or a general-purpose register that holds the operand (Col 26, line 65- Col 27, line 2. Operation routines are stored in memory.).

As per claim 10, Carron teaches the method of claim 1 wherein the branch instruction allows branches to occur based on evaluation of a byte that is in a data path of a processor (Col 134, lines 8-12. Branching occurs of byte in buffer.).

As per claim 13, Carron teaches the method of claim 1 wherein the branch instruction includes a Byte_spec Number that specifies the byte in the register to be compared with byte_compare_value (Col 39, lines 25-29; Col 134, lines 23-25. Instruction specifies the byte to be compared with the variable.).

Claims 4 and 5 are rejected under 35 U.S.C. 103(a) as being unpatentable over Carron, White, and Yates, in view of Atkins et al, US Patent #5,898,866 (Atkins hereinafter).

As per claims 4 and 5, Carron does not specifically teach the method of claim 1 wherein the branch instruction comprises: an optional token that is set by a programmer and specifies a number i of instructions to execute following the branch instruction before performing the branch operation where the number of instruction before performing the branch operation where the number of instructions can be specified as one, two, or three. Atkin teaches of an implementing hardware to execute loops for a branch

operation, wherein a field in the instruction specifies the number of instructions to execute prior to branching (Col 2, lines 28-31; Col 10, lines 21-23.).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teachings of Carron and Atkins with the motivation that the teachings of Atkin's to specific the number of instructions prior to branching would enhance the system of Carron, White, and Yates by reducing code space and decreasing execution time.

Claims 7, 8, and 11 are rejected under 35 U.S.C. 103(a) as being unpatentable over Carron, White, and Yates, in view of Bruckert et al, US Patent #4,742,151 (Bruckert hereinafter).

As per claims 7 and 8, Carron taught of instructions to execute following the branch instruction before performing the branch operation (Col 134, lines 8-26). However, Carron does not specifically teach the method wherein the branch instruction comprises: an optional token that is set by a programmer and which specifies a guess_branch prefetch for the instruction for the "branch taken" condition rather than the next sequential instruction. Brucket teaches of executing instructions where a determination is made as to whether a branch should or should not be taken, and teaches of prefetching "branch taken " instructions (Col 1, lines 22-29).

It would have been obvious to one of ordinary skill in the art the time the invention was made to combine the teachings of Carron and Brucket with the motivation that all the teachings deal with executing branching instructions in a processing system and the teachings of Brucket to prefetch "branch taken" instructions and executing branch guessing instructions would improve the system of Carron, White, and Yates by allowing high instruction throughput.

As per claim 11, Carron teaches the method of claim 1 wherein the branch instruction branches on a byte matching the byte value (Col 134, lines 8-10). However, Carron does not specifically teach wherein instruction prefetches the instruction for the "branch taken" condition. Brucket teaches of executing instructions where a determination is made as to whether a branch should or should not be taken, and teachings of prefetching "branch taken " instructions (Col 1, lines 22-29).

It would have been obvious to one of ordinary skill in the art the time the invention was made to combine the teachings of Carron and Brucket with the motivation that the teachings of Brucket to prefetch "branch taken" instructions and executing branch guessing instructions would improve the system of Carron, White, and Yates by allowing high instruction throughput.

Claim 9 is rejected under 35 U.S.C. 103(a) as being unpatentable over Carron, White, and Yates, in view of Gusefski et al, US Patent #5,202,972 (Gusefski hereinafter).

As per claim 9, Carron does not specifically teach the method of claim 1 wherein the branch instruction allows a programmer to specify which bit of the register to use to determine the branch operation. Gusefski teaches a computer system executing branch operations, where the hardware allows the selection of bits from the registers to determine the branch operation (Col 7, lines 33-36).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine teachings of Carron and Gusefski with the motivation that the teachings of Gusefski to allow for the selection of the bit of the register to use for the branch operation would improve the teachings of Carron, White, and Yates by increasing the capability of the programmer to control the branching processes.

Claim 12 is rejected under 35 U.S.C. 103(a) as being unpatentable over Carron, White, and Yates in view of Rodriguez, US Patent #6,139,199 (Rodriguez hereinafter).

As per claim 12, Carron does not specifically teach the method of claim 1 wherein the branch instruction branches on a byte not matching the byte value and wherein the instruction prefetches the next sequential instruction.

Yates teaches of performing a branch instruction if the byte contained in the register is not equal to a specified value (Col 77, lines 29-33).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teachings of Carron and Yates with the motivation that the teachings of Yates to execute branching when the bytes do not match would improve the system of Carron, White, and Yates by allowing for the continuation of executing instructions in conditional processing.

Rodriguez teaches of prefetching instructions (Col 12, lines 49-57).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teachings of Carron, White, Yates, and Rodriguez with the motivation that the teachings of Rodriguez to prefetch instructions would improve the system of Carron, White, and Yates by ensuring that the execution units are performing operations, which would reduce the time required to process instructions.

(10) Response to Argument

Appellant argued that:

(1) Claims 1-3, 6, 10, 13-21 are patentable over Carron, in view of Yates and White because the examiner has failed to show all of the features of claim 1 through the alleged combination of references,

in particular the feature of a "byte value specified by the branch instruction". Examiner acknowledges that Carron and White do not teach the particular feature and uses Yates. Yates teaches a compare instruction that specifies a value. Yates however uses a separate instruction that performs a branch.

In response, the appellant has misinterpreted examiner's rejection that the examiner acknowledged that Carron and White do not teach the particular feature. As set forth in the office action dated 5/2/2006, Carron teaches of a COMP_CHAR command, wherein the COMP_CHAR, "Compare[s] byte at specified position in the designated buffer with the byte constant in the command. Branch if the test passes." (Col 134, lines 29-34) Carron teaches of a command comprising instructions for compare and branch. However, Carron does not specifically teach of a single instruction for performing compare and branch. Therefore, the reference White was combined with Carron to specifically teach the feature, wherein White teaches of a compare-and-branch type instruction, which "combines compare and branch operations into one instruction."(Col 5, lines 26-29). Yates was used because Carron does not specifically teach the byte constant in the command is compared to a byte in the register. Yates teaches an instruction 884a that performs a byte compare of register to AL to the constant 3", and instruction 884b that "performs a branch if the value contained in the register AL is not equal to 3 (Col 77, lines 29-32). Even though Yates teaches of separate compare and branch instructions, White taught that compare and branch instructions can be combined into one instruction.

(2) There is no motivation for combining Carron, Yates, and White because Carron's COMP_BYTE, much like Carron's other commands, are not processor-executable instructions, and cannot include instructions that are unique to a particular processor. Carron's commands are provided in a high-level programming language that does not refer to architecture-specific operands such as registers. White's COBR instruction is a machine level instruction on a particular processing environment, and Yates' are machine-level instructions configured for executing in a particular processing environment.

The combination of Carron's high-level commands with Yates and White's machine-level instructions would render Carron unsatisfactory, and would teach away from Carron since Carron's high-level commands are intended to avoid using machine-level instructions.

In response, Carron teaches,

> "the general purpose operation routines will be created in machine language, which can be assembled into machine instructions which, in turn, can be directly executed by the central processor unit in the local terminal. This central processor unit will typically be a microprocessor having a specific machine instruction set. Utilizing machine language programming and having such programs authored by a highly skilled machine language programmer will provide the fastest execution time for individual general purpose operation routines and thus faster execution time for application programming running the local terminal. (Col 18, lines 40-52).
>
> "the method of this invention is based on storing in read only memory circuits within the local terminal a number of general purpose operation routines which comprise instructions to be executed by the central processor unit to accomplish a particular program task. Each of these general purpose operation routines is associated with a defined command. (Col 7, lines 59-66).

Carron teaches that the routines are processor-executable instructions, in which routines are associated with commands. The commands cause routines to be executed by a processor to accomplish a task, thus Carron's commands does not avoid machine-level instructions as asserted by the appellant. Furthermore, regardless of whether Yates and White's instructions are for particular processing environments, all three references teach similarly of executing instructions for comparing and branching, and the specific instructions of Yates and White would provide an enhancement to the commands taught by Carron.

In regards to the argument that there is no motivation for combining the references, the examiner recognizes that obviousness can only be established by combining or modifying the teachings of the prior art to produce the claimed invention where there is some teaching, suggestion, or motivation to do so found either in the references themselves or in the knowledge generally available to one of ordinary skill in the art. See *In re Fine*, 837 F.2d 1071, 5 USPQ2d 1596 (Fed. Cir. 1988)and *In re Jones*, 958 F.2d 347, 21 USPQ2d 1941 (Fed. Cir. 1992).

In this case, it would have been obvious to one of ordinary skill in the art at the time the invention was made to modify the teachings of Carron with the teachings of White to implement a compare-and-branch instruction, i.e. combine compare and branch operations into one instruction, with the motivation that the teachings of White would improve the teachings of Carron by reducing latency and providing efficiency for executing instructions (White, Col 2, lines 1-34). Furthermore, it would have been obvious to one of ordinary skill in the art to combine the teachings of Carron, White, and Yates because all three teachings with the similar instructions of compare and branch. Furthermore, the teachings of Yates to compare a value in the register would improve the system of Carron and White with the motivation that the Yates' teachings would enhance the ability of a processor by allowing the processor to execute different types of instructions to perform a task since a register, an high-speed storage area within the CPU, would provide values that are used for performing instructions, such as adding.

(3) Regarding dependent claim 4, examiner's characterization of Atkins is incorrect. Atkins does not teach or disclose at least, "wherein the branch instruction comprises: an optional token that is set by a programmer and specifies a number i of instructions to execute following the branch instruction before performing the branch operation."

In response, the examiner stated in the office action dated 3/17/2006, that "Atkins teaches of implementing hardware to execute loops for a branch, wherein a field in the instruction specifies the number of instructions to execute prior to branch." Atkin's specification recites that, "[a] instruction has a length field which specifies the number of instructions within the loop" (Col 2, lines 28-31), and "It is decremented with each iteration to control the branch at the bottom of the loop. BOT is the number of instructions within the loop (i.e. two)." (Col 10, lines 1-3). The examiner's representation of Atkin's is clearly a reasonable interpretation and within the intended meaning of the reference, and thus not a mischaracterization of Atkins.

Carron teaches of executing a branch command and performing a branch operation (Col 134, lines 28-34). Carron does not specifically teach the feature of specifying a number i of instructions to execute following the branch instruction before the branch operation wherein the number of instructions can be specified as one, two, or three. But Atkin's teaches of programming to specify a number of instructions to execute. It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teachings of Carron of executing a branch command and performing a branching operation with the teachings of Atkin of specifying a number of instructions, thus to execute a branch command specifying a number of instructions before performing a branch operation because Atkin's teachings would enhance Carron's teaching by allowing a programmer to modify an instruction, i.e. specify a number of instructions, to perform different required operations, i.e. changeable coding, and reducing the number of instructions, which would speed execution (Atkin, Col 2, lines 13-19).

(4) Regarding dependent claim 7, Brucket describes a processor that fetches two instructions when the processor encounters a branch instruction: one instruction corresponding to the "branch taken" eventuality as well as the "branch not taken". Neither of these eventualities correspond to the claimed branch guess token. Because both instructions are retrieved, Brucket would have no need to include a token that would indicate which instruction should be retrieved following the execution of a branch instruction.

In response, Brucket teaches of determining whether a branch should be taken depending on prior processing and prefetching the "branch taken" instruction and "branch not taken" instruction (Col 1, lines 22-29). Brucket does not explicitly teach of a "token" for specifying a prefetch for the "branch taken" instruction. However, an indicator or an identifier such as a token would still be needed to indicate the retrieval of the "branch taken" instruction. Furthermore, the prefetch for "branch taken" or the indication for prefetch, e.g. indicator, identifier, or token, is optional in that a programmer has the option to include

the specific prefetch as part of the programming. Appellant's "optional token" in the claim does not define what is optional about the token such that the token is selected over another token.

(5) Regarding dependent claim 8, Carron does not disclose or suggest an optional token that specifies a number i of instructions to execute following a branch instruction and a second optional token that is set by a programmer and which specifies a guess_branch prefetch for the instruction for the "branch taken" condition rather than the next sequential instruction. Carron does not disclose or suggest anywhere that its COMP_BYTE command performs any type of branch defer operation that causes the execution of additional instructions following the branch instruction and before the branch operation is performed.

In response, firstly, claim 8 does not specifically define the argued feature of a "branch defer operation". Therefore, it is not clear as to what appellant's argued "branch defer operation" is specifically referring to in the appeal brief. It appears appellant is equating the branch defer operation with the "guess_branch prefetch" and will be interpreted as such. Claim 8 also does not define a feature of performing a branch defer operation that causes the execution of additional instructions following the branch instruction and before the branch operation. Claim 8 recites, in additional to other features, "a second optional token... which specifies a guess_branch prefetch for the instruction for the "branch taken"," but the claim does not define the feature that the guess_branch prefetch causes the execution of additional instructions nor that the guess_branch prefetch is following the branch instruction and before the branch operation. It is noted that the features upon which appellant relies are not recited in the rejected claim(s). Although the claims are interpreted in light of the specification, limitations from the specification are not read into the claims. See *In re Van Geuns*, 988 F.2d 1181, 26 USPQ2d 1057 (Fed. Cir. 1993).

Carron teaches of the COMP_BYTE command or the COMP_CHAR command, commands that invoke instructions for executing a branch operation (Col 134, lines 14-26, 37-49), to which appellant also stated in appellant's Remarks dated 12/29/2005, page 9, "Carron's COMP_BYTE, much like Carron's other commands... invokes a routine comprising several instructions stored in memory." Therefore, when creating the command, a programmer may specifiy the number of instructions, i.e. how many or which instructions, to execute prior to the branch operation. Carron further teaches of a command comprising, "Get the number of instructions on the application input circular list." and "Decrement the number of instructions remaining." (Col 592, lines 32-35, 48-52)

As set forth in the office action dated 3/17/2006, Carron does not specifically teach, "a second optional token that is set by a programmer and which specifies a guess_branch prefetch for the instruction for the "branch taken" condition rather than the next sequential instruction. Brucket teaches of determining whether a branch should be taken depending on prior processing and prefetching the "branch taken" instruction and "branch not taken" instruction (Col 1, lines 22-29). Brucket does not explicitly teach of a "token" for specifying a prefetch for the "branch taken" instruction. However, a command or an identifier such as a token would be needed to indicate the prefetch for the "branch taken" instruction. Furthermore, the prefetch for "branch taken" or the indication for the prefetch, e.g. command, identifier, or token, is optional in that a programmer has the option to include the specific prefetch as part of the programming. Appellant's feature of optional token in the claim does not define what is optional about the token.

For the above reasons, it is believed that the rejections should be sustained.

Respectfully submitted,

JJ

November 8, 2006

Conferees:

Nathan J. Flynn N.J.F.

Joshua Joo

Lynne H. Browne
Appeal Specialist, TQAS
Technology Center 2100